CSC 282 - Fall 2011

http://www.cs.rochester.edu/~stefanko/Teaching/11CS282/

BASIC PART (4 pages, 80 points)

1. (20 POINTS) We are given n numbers $a_1, \ldots, a_n$ (they can be positive or negative). A selection S of the numbers is called **valid** if

   no three consecutive numbers are selected in S.

   The goal is to find a valid selection of the numbers with the maximal sum. We are going to give an $O(n)$-time dynamic programming algorithm for the problem. We will have a table $P[1..n]$ where

   $$P[i] = \text{the maximal sum of a valid selection of the first } i \text{ numbers.}$$

   Give an expression (or a piece of code) for $P[i]$ in terms of $P[i-3]$, $P[i-2]$, $P[i-1]$, a_i, and a_{i-1}.

   $P[i] =$

2. (20 POINTS) We are given n numbers $a_1, \ldots, a_n$ (they can be positive or negative). A selection S of the numbers is called **valid** if

out of each three consecutive numbers at least one is selected.

The goal is to find a valid selection of the numbers with the minimum sum. We are going to give an $O(n)$-time dynamic programming algorithm for the problem. We will have a table $P[1..n]$ where

$$P[i] = \begin{array}{l}\text{the minimum sum of a valid selection } S \text{ of the first } i \text{ numbers,} \\ \text{with the constraint that } i \text{ is selected (that is } i \in S).\end{array}$$

Give an expression (or a piece of code) for $P[i]$ in terms of $P[i-3]$, $P[i-2]$, $P[i-1]$, and a_i.

$P[i] =$

3. (20 POINTS) Let $X = x_1, \ldots, x_m$, $Y = y_1, \ldots, y_n$ be two sequences. Our goal is to compute the length of the longest common subsequence[1] of X and Y. We will have a 2-dimensional table $T[0..m, 0..n]$, where

$$T[i,j] = \text{the length of the longest common subsequence of } x_1, \ldots, x_i \text{ and } y_1, \ldots, y_j.$$

Give an expression (or a piece of code) to compute $T[i,j]$ from the previously computed values of T (of course you will also need the values in X and Y in your expression).

[1] that is the largest ℓ such that there exist $1 \le i_1 < \cdots < i_\ell \le m$ and $1 \le j_1 < \cdots < j_\ell \le n$ such that $x_{i_k} = y_{j_k}$ (for all $k \in \{1, \ldots, \ell\}$).

4. (20 POINTS) Let $X = x_1, \ldots, x_m$, $Y = y_1, \ldots, y_n$ be two sequences. Our goal is to compute the length of the longest common substring[2] of X and Y. We will have a 2-dimensional table $T[0..m, 0..n]$, where

$$T[i,j] = \text{the length of the longest common substring of } x_1, \ldots, x_i \text{ and } y_1, \ldots, y_j \text{ which ends with } x_i \text{ and } y_j \text{ (in the notation from the footnote } s + \ell - 1 = i \text{ and } t + \ell - 1 = j).$$

Give an expression (or a piece of code) to compute $T[i,j]$ from the previously computed values of T (of course you will also need the values in X and Y in your expression).

[2] that is the largest ℓ such that there exist $s \in \{1, \ldots, m+1-\ell\}$ and $t \in \{1, \ldots, n+1-\ell\}$ such that $x_{s+k} = y_{t+k}$ (for all $k \in \{0, \ldots, \ell-1\}$).

ADVANCED PART (2 pages, 40 points)

1. (20 POINTS) We are given a sequence A of n numbers $a_1, \ldots, a_n$. We would like to count the number of different subsequences of A. For example, $A = 1, 2, 3$ has 8 different subsequences (one of length 3, three of length 2, three of length 1, and one of length 0); on the other hand $A = 1, 1, 1$ has 4 different subsequences (one of length 3, one of length 2, one of length 1, and one of length 0). Give a dynamic-programming algorithm for the problem. Clearly describe the array you are using and how you compute the array. Pseudocode is required for this problem.

2. (20 POINTS) We have a 25×25 chessboard filled with numbers (each square has one number). We want select a subset of the squares so that 1) their total sum is maximized, and 2) there is no 2×2 square which has all of its squares selected. Give an algorithm for the problem, argue why is it correct, and state the running time of your algorithm. Pseudocode is not required but you should give a very clear description of your algorithm. The running time of your algorithm should be such that when implemented on today's machines it will finish in less than a minute.